A Case for Heterogeneous On-Chip Interconnects for CMPs

Asit K. Mishra N. Vijaykrishnan Chita R. Das
Department of Computer Science and Engineering
The Pennsylvania State University
{amishra, vijay, das}@cse.psu.edu

ABSTRACT

Network-on-chip (NoC) has become a critical shared resource in the emerging Chip Multiprocessor (CMP) era. Most prior NoC designs have used the same type of router across the entire network. While this homogeneous network design eases the burden on a network designer, partitioning the resources equally among all routers across the network does not lead to optimal resource usage, and hence, affects the performance-power envelope. In this work, we propose to apportion the resources in an NoC to leverage the non-uniformity in network resource demand. Our proposal includes partitioning the network resources, specifically buffers and links, in an optimal manner. This approach results in redistributing resources such that routers that require more resources are allocated more buffers and wider links compared to routers demanding fewer resources. This results in a novel heterogeneous network, called HeteroNoC, which is composed of two types of routers – *small* power efficient routers, and *big* high performance routers. We evaluate a number of heterogeneous network configurations, composed of big and small routers, and show that giving more resources to routers along the diagonals in a mesh network provides maximum benefits in terms of performance and power. We also show the potential benefits of the HeteroNoC design by co-evaluating it with memory-controllers and configuring it with an asymmetric CMP consisting of heterogeneous cores.

Categories and Subject Descriptors

C.1.2 [**Computer Systems Organization**]: Multiprocessors; Interconnection architectures; C.1.3 [**Other Architecture Styles**]: Heterogeneous (hybrid) systems

General Terms

Design, Experimentation, Performance.

1. INTRODUCTION

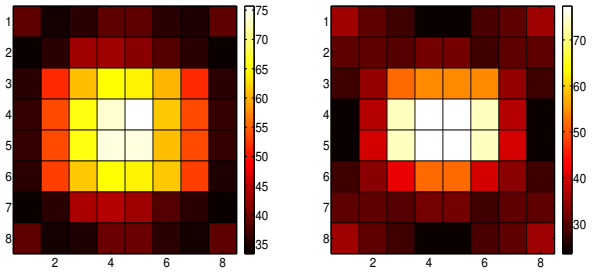
Multicore architectures seem to be the only plausible solution to meet the performance and power requirements of a wide variety of applications targeted for the general-purpose CMP and the special purpose SoC environments. As per the ITRS road map [1], it is projected that the performance demand would continue to grow to 300x by 2022, which in turn would require chips with 100x more cores than the current state-of-the-art design. This upward performance trend coupled with the need to minimize chip power consumption necessitates a fresh look at the design of future multicores. The three main components of a multicore that dominate the performance and power envelope are the processor cores, memory, and the underlying on-chip interconnect or Network-on-Chip (NoC) architecture. It is known that in addition to orchestrating the system performance, an on-chip network can consume up to 28% of the chip power [12], and thus, design of high performance and low-power interconnects is essential for sustaining the multicore growth in coming years.

An $N \times N$ mesh interconnect is the most widely studied NoC topology due to its scalability, regularity and ease of implementation in silicon. While different variations of the mesh architecture have been proposed recently [4, 7, 17], all of them are centered around the same homogeneous router design, where all routers are provisioned with the same silicon real estate. On the contrary, it is known that the center of a mesh usually gets more congested because it handles more traffic compared to the edge routers with deterministic X-Y routing [6]. This is demonstrated in Figure 1 for an 8x8 network with the uniform random (UR) traffic pattern[1]. Figures 1 (a) and (b) show on a heat-map scale the average buffer and link utilization. We observe that, while the routers in the center of the mesh are highly ($\backsim$75%) utilized , the peripheral routers have low ($\backsim$35%) utilization, and the routers around the center of the mesh have buffer utilization that lie between these two extremes. A similar trend is seen for average link utilization as depicted in Figures 1(b). This behavior is consistent at both medium and high network loads, and for a wide variety of traffic patterns. Moreover, the routers at the corners of the mesh, show slightly higher buffer and link utilization than the rest of the routers in the same rows and columns.

Additionally, our analysis shows that the non-uniformity in resource utilization is an artifact of any non-edge symmetric network employing deterministic X-Y routing. For instance, concentrated mesh and flattened butterfly topologies are not edge symmetric, and exhibit non-uniform resource usage as well. Figure 2 shows the buffer utilizations in a 4x4 concentrated mesh with concentration degree of 4 and in a 64 node flattened butterfly [15] topology (each router is connected to 4 nodes; thus, 16 routers) with

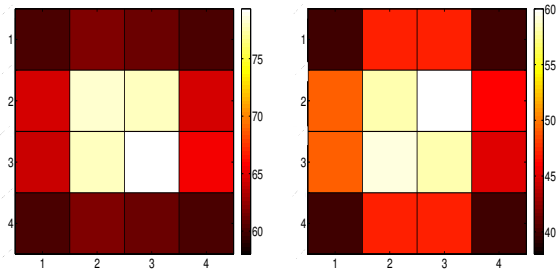
[1]The network is wormhole switched, uses deterministic X-Y routing and is operated at close to saturation throughput (6% packets/node/cycle). This baseline network has 3 virtual channels per physical channel, 5-flit buffer depth with 192 bit flit-width.

ISCA'11, June 4–8, 2011, San Jose, California, USA.



(a) Buffer utilization (b) Link utilization

Figure 1: Buffer and link utilization (in percentage) across all routers in an 8x8 mesh on a heat-map scale.



(a) Concentrated Mesh (b) Flattened Butterfly

Figure 2: Buffer utilization (in percentage) in other topologies on a heat-map scale.

UR traffic. Like the mesh topology, these two topologies also exhibit non-uniform resource demand (link utilization results are not shown due to brevity.), and thus, this argument should be true for any non-edge symmetric network, although we use the mesh interconnect to demonstrate our idea in this paper.

This observation of non-uniform resource usage leads us to rethink the traditional mesh interconnect design by questioning the rationality of uniform resource distribution across all routers. Intuitively, since the central routers in a mesh handle more traffic compared to the peripheral routers, we should be able to enhance performance by allocating more buffer and link bandwidth to the center routers compared to the latter. This implies that we should be looking at different types of router architectures, where the peripheral routers employ a small router with fewer buffers (virtual channels (VCs)) and narrow link width, while the central and other highly utilized routers employ relatively large number of VCs and wider link width (*big* routers). This essentially moves us from the homogeneous NoC to a heterogeneous NoC domain, an emerging area that has remained unexplored in the context of performance-power trade-offs.

The heterogeneous NoC concept is further supported by the fact that future multicores/SoCs will have heterogeneous cores and components [11, 18, 19]. However, it is not intuitively clear if such heterogeneous cores would benefit more from a heterogeneous NoC. Towards this end, this paper examines the rationality of resource redistribution and proposes a heterogeneous mesh architecture, called HeteroNoC, for optimizing both performance and power. In this context, we attempt to answer the following questions: *(i) Should we design a heterogeneous NoC with two types (small and big) of routers? (ii) If yes, how many such big/small routers do we need and how do we redistribute the buffer and link width between these routers without changing the original bisection width and buffer resources? (iii) Is there an optimal placement of big routers that would maximize the performance and power benefits compared to the baseline homogeneous mesh? and (iv) How else can a HeteroNoC design be leveraged to achieve better performance/power in a CMP?*

Starting with an $(N \times N)$ homogeneous mesh network, we explore the design space of the proposed HeteroNoC with two types of routers with different link widths and buffer organizations, and placement of such routers, while keeping the bisection bandwidth the same. Our analysis with synthetic and real applications shows that with *small* and *big* routers, where the big routers are assigned more VCs and double the link width of small routers, we can design a heterogeneous mesh with $2N$ big routers for providing better performance and power behavior over the baseline homogeneous network. Furthermore, by placing the big routers in the diagonals of a mesh to help traffic flow in high activity network regions, we get the maximum benefit compared to other placements. The advantages of the HeteroNoc architecture is also demonstrated in the context of heterogeneous CMPs and placement of memory controllers. In summary, the primary contributions of this paper are the following:

• We propose a novel heterogeneous network for general purpose CMPs and show that, strategic placement of big and small routers connected to homogeneous cores can help improve performance and reduce power. Using the *same* amount of link resources and 33% *fewer* buffer resources compared to a homogeneous network, we show that a carefully designed heterogeneous network can reduce average latency by 23%, improve network throughput by 24% and reduce power by 26% with synthetic traffic patterns. With real applications, the best HeteroNoC design shows 18.5% reduction in packet latency leading to 12% improvement in IPC, while consuming 22% less power compared to an equivalent homogeneous network. Moreover, we show that these benefits are realized using a very simple implementation that does not require significant modification to the router design.

• After a rigorous design space exploration, we observe six heterogeneous network layouts that provide superior performance among all possible placements, and show detailed performance/power characterization with each one of them to conclude that the diagonal placement of big routers results in the best configuration.

• To further demonstrate the applicability of our HeteroNoC proposals, we co-evaluate our proposed network with a recently proposed memory controller placement scheme [2], and show how the presence of big and small routers in a network benefits the placement of memory controllers. Furthermore, we also show that heterogeneous CMPs, composed of large and small cores, can leverage a heterogeneous network to further improve system performance.

The remainder of this paper is organized as follows: In Sections 2 and 3, we discuss the architectural and micro-architectural design details for the HeteroNoC, respectively; Section 4 summarizes our experimental platform and in Sections 5, 6 and 7, we discuss the performance and power benefits. Related work is discussed in Section 8, followed by conclusions in Section 9.

2. HETERONOC ARCHITECTURE

Without loss of generality, we chose a (5x5) baseline router with 3VCs per physical channel (PC), and 192b links/flit-width. Table 1 shows the architectural level details of the homogeneous routers. The baseline (8x8) homogeneous network is configured using these routers. For keeping the design space of the HeteroNoC simple, we considered the network to be composed of *two* types of routers - *small* power efficient routers, and relatively *big* performance boosting routers. However, a network designer might choose other types of routers to compose a heterogeneous network at the expense of

Table 1: Comparison of homogeneous and heterogeneous routers

		Power	Area	Frequency
Homogeneous network	3VCs/5 buffer depth/192b	0.67W	$0.290mm^2$	2.2 GHz
Heterogeneous network	2VCs/5 buffer depth/128b (small router)	0.30W	$0.235mm^2$	2.25 GHz
	6VCs/5 buffer depth/256b (big router)	1.19W	$0.425mm^2$	2.07 GHz
Total buffers in **homogeneous network** = 64 (routers) * 3 (VCs) * 5 (PCs) * 5 (buffer depth) = 4800 buffers @ 192 bits/buffer = 921,600 bits				
Total buffers in **heterogeneous network** = 48 (routers) * 2 (VCs) * 5 (PCs) * 5 (buffer depth) + 16 (routers) * 6 (VCs) * 5 (PCs) * 5 (buffer depth) = 4800 buffers @ 128 bits/buffer = 614,400 bits (33% reduction over the homogeneous case)				

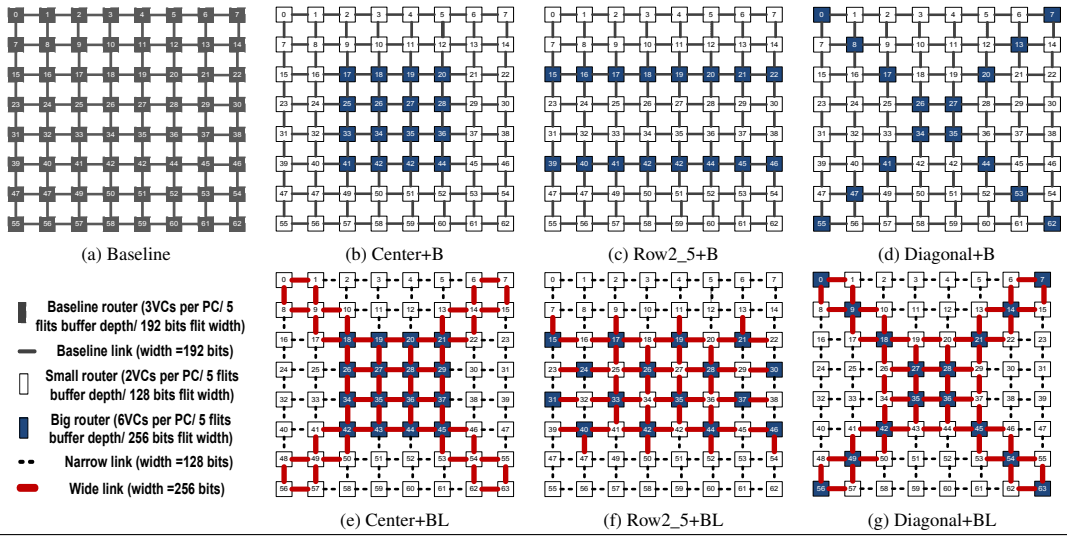


Figure 3: Six layouts of an (8x8) HeteroNoC

added complexity in design and verification. Since buffers and links are the primary building blocks of a (5x5) router, we considered redistributing these resources to leverage non-uniform resource utilization in the network. There were two primary constraints in designing the small and big routers - *the total number of VCs and network bi-section bandwidth are kept the same in the baseline and heterogeneous networks*.

HeteroNoC Link Re-distribution: Since we decided to use only two types of routers, a natural choice was to have two kinds of links connecting these routers. Link width significantly affects router power (precisely buffer and crossbar power), and since our goal is to design power efficient small routers, their crossbar width (and hence, link width) should be less than that of big routers. This design principle and constant bisection bandwidth constraint led us formulate the link width equation as:

$W_{homo} \times n = W_{hetero} \times N_{narrow} + 2W_{hetero} \times N_{wide}$;

where W_{homo} is the link width in the homogeneous network, n is the number of links crossing the bisection-cut in one direction in the homogeneous network, W_{hetero} is the link width of small routers, and N_{narrow} and N_{wide} are the number of narrow and wide links, respectively, crossing the bisection-cut in the heterogeneous network.

In our case, with an (8x8) baseline network with 192b links, and considering an equal size (= 8x8) heterogeneous network bisection-cut to consist of equal number of wide and narrow links[2], we have: $192 \times 8 = W_{hetero} \times 4 + 2W_{hetero} \times 4$; implying $W_{hetero} = 128$. Hence, we chose narrow links to be 128b and wide links to be 256b.

One other design guideline that dictated the wide-link width to be double of the narrow-link width is that, narrower links (compared to baseline) in small routers dictate the flit-width to be less than the baseline network (128b vs. 192b), and hence, transfer latency increases in the heterogeneous network. We use a novel scheme to merge two flits together and transmit them simultaneously for reducing the transfer latency (discussed later in Section 3.2). This simultaneous transmission of two flits, requires the link width of big routers to be double of the flit-width.

HeteroNoC Buffer Re-distribution: Since buffers consume about 35% of router power [29, 30], having more VCs in a router increases the network power consumption. However, more buffers (and hence, VCs) in a router improve performance. In our design, we allocate more VCs to big routers and fewer VCs to small routers by stripping a few VCs from the small routers, while keeping the total number of VCs constant with respect to the homogeneous network. Our experimental analysis shows that, having 1VC/PC in a router degrades performance due to the inability of the router in multiplexing packet flows. Hence, we provision 2VCs/PC in the small routers and allocate 3 VCs stripped from three baseline routers (to make them small routers) to another baseline router to make it a big router with 6VCs/PC. The power and area profiles of the big and small routers are shown in Table 1.

Number of Small and Big Routers: Our design goal was to keep the network area and power consumption of the heterogeneous network less (or equal) than that of the homogeneous network. This constrained the number of big routers in the heterogeneous network. To calculate the number of small and big routers in the network, we use the inequality:

[2]Compared to the baseline design consisting of 8 192b links crossing one direction, our initial evaluations showed significant performance benefit with just equal number (4) of narrow and wide links. A sensitivity study of how performance varies with the change in ratio of the number of wide and narrow links is left for future work.

$0.67 \times N^2 \geq 0.3 \times n_s + 1.19 \times (N^2 - n_s)$; where 0.67, 0.3 and 1.19 are the power consumption (in Watts) of the baseline, small and big router respectively, N^2 is the total number of routers in the mesh network and n_s is the number of small routers[3]. Simplifying the inequality gives us $1.71 \geq \frac{N^2}{n_s}$.

In our design, $N = 8$, implying $n_s \geq 37.4$. Thus, we should have a minimum of 38 small routers in the network to guarantee that the heterogeneous network is power efficient than the homogeneous network. Symmetry considerations led us select 48 small routers and 16 (= 2N) big routers, where for every 3 small routers, we have a big router in the network (this was the reason for stripping a VC from three baseline routers to make them small routers and allocating the 3 VCs to a big router).

Placement of Routers in HeteroNoC: The next obvious question is how do we place these two types of routers in the HeteroNoC. Here, we consider two design options. First, let us design the big and small router by redistributing the buffers only (6VCs/PC and 2VCs/PC for big and small routers, respectively) without changing the original link width (192b). Next, we redistribute both the buffer and link width for the two routers (6VC/PC and 256b link width for big routers, and 2VCs/PC and 128b width for small routers). Figure 3 shows six layouts considered for our evaluations. We annotate the buffer only redistribution configurations with a +B extension and combined buffer with link redistribution with a +BL extension after the names of the configurations. Figure 3 (a) shows the baseline network layout and Figures 3 (b), (c) and (d) show the three network layouts with only buffer re-distribution. Figures 3 (e), (f) and (g) depict the HeteroNoC layouts with combined buffer and link redistribution. These heterogeneous configurations were the best six configurations among *thousands* of configurations evaluated, and hence, for experimental evaluations, we only discuss these six layouts[4].

The **Center+B** (Figure 3 (b)) layout is motivated by non-uniform buffer utilization (Figure 1) in the network, where we apportion more buffers in the central regions of the network and reduce buffers from the peripheral routers. In the **Row2_5+B** layout (Figure 3 (c)), the big routers are arranged in the second and fifth row of the mesh network for minimizing the average hop count to reach a big router. In the **Diagonal+B** configuration, the big routers are arranged along the network diagonals to spread the big routers as far as possible and also to have a few big routers in the center of the mesh network. Particularly, placing a few big routers in each row and column helps most of the flows to use the big routers. Also, big routers at the corner of the mesh network help alleviate resource contention in these routers.

With both the buffer and link re-distribution approach, we allocate more buffers and links to the highly utilized routers in the network. Again, we consider the above three configurations to evaluate the combined buffer and link redistribution approach. **Center+BL**, **Row2_5+BL** and **Diagonal+BL** are similar to the Center+B, Row2_5+B and Diagonal+B layouts, respectively, with the exception that the big routers have wider links (256b) and small routers have narrow links (128b).

[3]This inequality serves as an empirical guideline. The power profile of each router changes with its utilization levels, and the numbers present in the inequality represent the power profile of a router at 50% activity factor. However, in all our simulations, we use the actual utilization of a router to calculate its power consumption.

[4]Apart from the heterogeneous network configurations presented, we did a comprehensive design space exploration on a smaller network size and extrapolated the configurations to an 8x8 network size. Specifically, with a 4x4 network, we evaluated all possible placements of *(4 small, 12 big)*, *(6 small, 10 big)* and *(8 small, 8 big)* routers. In our case, the number of simulations was tractable since, symmetry and constant bisection bandwidth constraints limited the number of cases to be evaluated to a few thousands (1820, 8008 and 12870 configurations in each case respectively). Such an exhaustive analysis in an 8x8 network is infeasible because the design space bloats (e.g. the number of ways to place 48 small routers and 16 small routers in a 64 node network is $\binom{64}{48}$ = 4.89E+14).

3. HETERONOC DESIGN DETAILS

3.1 Impact on Crossbar Design

HeteroNoC design requires minor modifications to the buffering, switch arbitration and crossbar stages of the routers. In the HeteroNoC architecture with combined buffer and link re-distribution, we use both 128b and 256b links with flit size being 128b. Hence, when communication takes place between a small and a big router (or two big routers) between which a 256b link exists, two 128b flits can be combined to be simultaneously sent over the wider link. This is depicted in Figures 4 (b)-(f), where we show the possible crossbar architectures for HeteroNoCs. Essentially, in all the HeteroNoC configurations, when a small and big router communicate, rather than having a single input port mapped to a single output port, there exists a possibility of two input ports to be able to map to a single output port so that flits from these input ports can simultaneously be sent to the output.

3.2 Impact on Buffer Read/Write Stage

In our design, a 256b link exists between a small router and a big router, and between two big routers. Figure 5 (b) shows the modifications done to the buffering stage in the router compared to a standard design shown in Figure 5 (a), where B is the channel width. Our design is similar to the XShare technique proposed in [7], where two *small* flits (e.g. coherence traffic whose flit sizes are very small compared to data flits) are combined and sent over the link to the next router. However, in HeteroNoC, whenever possible, we combine two 128 data flits and transmit them simultaneously over the wider link. Simultaneous transmission of two flits requires minimal modification to the credit based flow control scheme - instead of requiring a credit for a single flit in the upstream router, the downstream router now needs two credits in the upstream router.

When two combined 128 bits arrive at an upstream router port, the individual flits are demuxed based on the virtual channel ids (VCIDs) of the two flits. This requires that the data path of the input DEMUX and the switch MUX to be split into two separable halves. Similarly, to combine two flits, a switch allocation (SA) control logic now sends two demux signals that determine which half of the crossbar a flit needs to be routed (Upper Half is Data Set1 (DSET1) and Lower Half is Data Set2 (DSET2)). The buffers are still maintained as 128 bit FIFO buffers requiring no additional modifications. They are treated as two separate logical halves (DSETs) by the MUX/DEMUX logic. The primary overhead comes from including the second layer of smaller muxes (shaded black in the figures). Typically, the buffer read / write stages are the shortest stages in a generic router pipeline, and hence, have sufficient slack for two 2:1 muxes without affecting the router cycle time.

3.3 Impact on SA Stage

Figure 6 (a) shows the switch arbitration (SA) stage of the baseline router. The SA stage can be broken into two sub-stages. In the first sub-stage, (SA stage 1), a v:1 arbiter for each input port chooses which of its virtual channels (v_i) can bid for an output port. In the second stage, (SA stage 2), a p:1 arbiter for each output port (p_o) chooses one of the input ports(p_i). Hence, at the end of

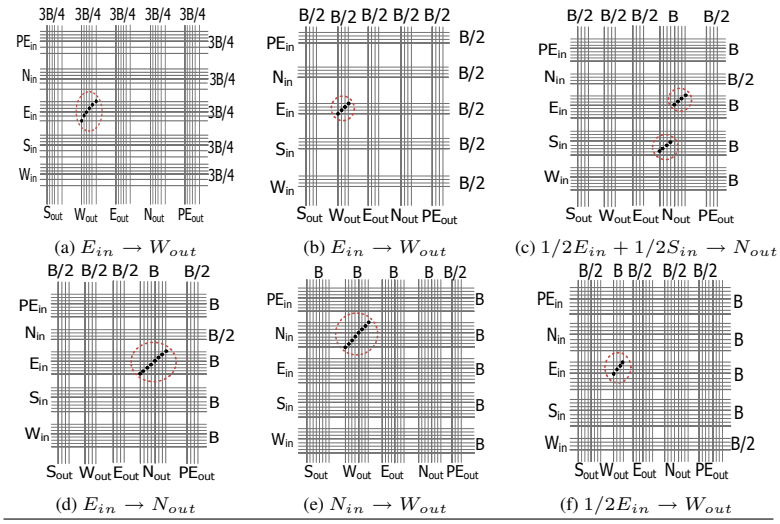


Figure 4: Crossbar architecture details (B=256 bits). The figure shows in each case a mapping (shown by the highlighted circle) from i/p port(s) to an o/p port; (a) Baseline xbar (192b wide); (b) Small router (128b wide) when connected to small routers on all sides e.g. router 11 in Fig. 3(g); (c) Small router when connected to a big router on N_{out} (shows two VC's from two different i/p ports of small router being mapped to N_{out}) e.g. router 52 in Fig. 3(f); (d) Small router when connected to a big router on N_{out} (shows two VC's from the same i/p port of small routers being mapped to an o/p port) (e) Big router (256b wide) when connected to big routers on all sides e.g. router 35 in Fig. 3(e) and; (f) Big router when connected to a small router on W_{out} e.g. router 26 in Fig. 3(e).

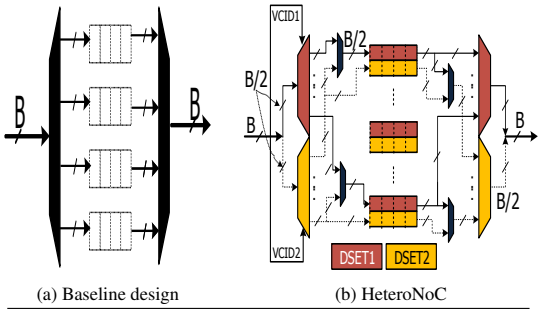


Figure 5: Baseline and HeteroNoC input buffer organization

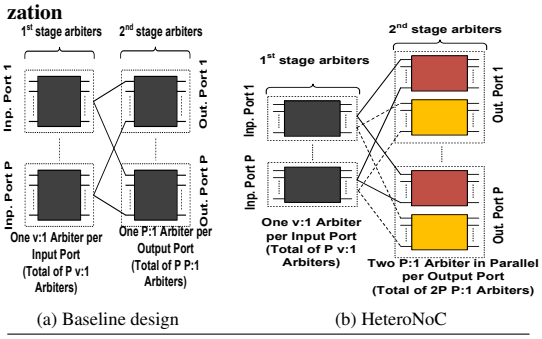


Figure 6: Baseline and HeteroNoC SA stage organization

the SA stage, a (p_i, v_i) pair is chosen for each output port p_o and a crossbar mapping is enabled between p_i and p_o. Two 128 bit flits can be combined in the HeteroNoC only when - (a) Two input VCs within a *single* input port request the *same* output port and (b) Two input VCs from *different* input ports request the *same* output port. In case (a), a combined request for both input VCs can be sent to SA stage 1, and if the combined request wins in the SA stage 2, both the flits can be sent together. In case (b), the requests can be combined in SA stage 2. For low network loads, analysis shows that, two flits can be combined 40% of the time, and at moderate to high network loads, this can be done 80% of the time.

The complexity of flit combination can be attributed to selecting the second flit in SA stage. This can be done by using two simultaneous p:1 arbiters as shown in Figure 6 (b). When the top arbiter picks a 128 bit, the second arbiter will supply a matching 128 bit flit if there exists one. The area overhead of additional arbiters is around 2.5% of the router area (obtained from Synopsys synthesis).

3.4 Impact on Router Frequency

A critical parameter that is affected as a result of increasing the buffers and the crossbar width in larger routers is the router operating frequency. Increasing the number of VCs in a router increases the cycle time, and hence, reduces the router frequency [22]. In our router design, the VA stage is the dominating stage and the frequency of the big routers consisting of 6 VCs is reduced by 6% compared to the frequency of the baseline router (=2.2 GHz) with 3 VCs. On the other hand, the frequency of the small routers with 2 VCs is boosted by 2% compared to the baseline frequency. For simplicity, in all our analysis, we consider the heterogeneous network to be operated at the worst case operating frequency, i.e. frequency of the big routers.

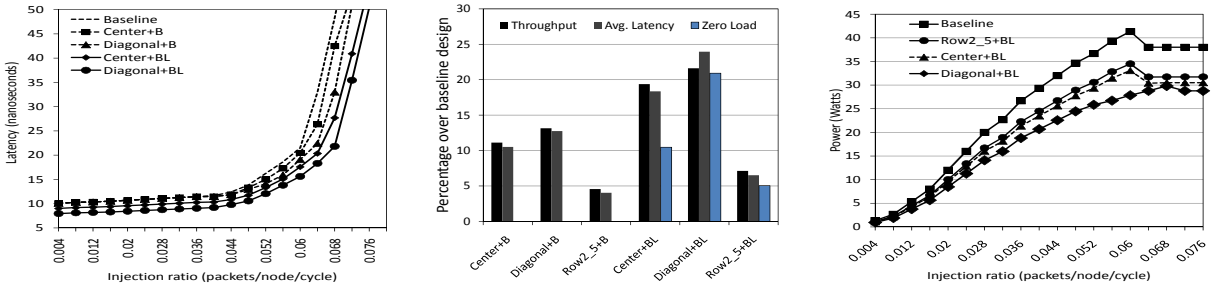
3.5 Impact on Area

Our analysis, using Synopsys Design Compiler, shows that the area of the big routers increase by 46% and the area of the small routers decreases by 18% compared to the baseline design. We believe such heterogeneous router layouts can be handled with floorplanning constraints that combine multiple processors and router blocks to create larger tiles. Hence, we assume that the increase in area of a router can be accommodated within the *tile* width of the core to which the router is connected and, thus, it does not increase the cycle time of the link stage. In fact, factoring out link area (since they are the same in both designs), the total area of HeteroNoC routers is 18.08 mm^2 (=0.235 mm^2*48 + 0.425mm^2*16) which is less than the total router area of the homogeneous network (= 18.56mm^2 = 0.290mm^2*64 routers).

4. EXPERIMENTAL SETUP

We conduct our experiments with a 64-node network, laid out as an 8x8 2D-mesh. We use a cycle-accurate NoC simulator and model a state-of-the art two stage router pipeline based on [27]. The baseline router in our case has 5 PCs including the local node-to-router port and 3 VCs/PC. For the HeteroNoC experiments, the number of VCs vary depending upon the router type. A data packet consists of 1024b (= cache line size) and is decomposed into 6 flits in the baseline design (with 192b buffer/crossbar/link width) and 8 flits in the HeteroNoC design (with 128b buffer/crossbar width). An address packet is composed of 1 flit in both cases. We use a buffer depth of 5 flits per VC across all designs.

We simulate a wormhole-switched network with deterministic X-Y routing and credit-based flow control. The router along with the proposed modifications, described in Section 3, was implemented in structural RTL Verilog and synthesized using Synopsys Design Compiler using 65 nm cell library. The homogeneous net-



(a) Latency (b) Throughput improvement and latency reduction (c) Power

Figure 7: Performance and network power behavior with UR traffic.

work operates at a clock voltage of 1V and 2.20 GHz and the HeteroNoC at 2.07 GHz. The dynamic and leakage power numbers were extracted using Orion [30] and incorporated in our simulator for detailed power analysis of the network. We report results for the seven configurations shown in Figure 3.

We use both synthetic and real world benchmarks for performance and power consumption analysis. When simulating synthetic traffic, the network is initially warmed up with 1000 packets and the statistics are collected for 100,000 packets. We measure average flit latency, average power consumption, and network throughput as a function of input load with Uniform Random (UR), Nearest Neighbor (NN), Transpose, Bit-Complement and Self-Similar traffic patterns. For real application workloads, we use four commercial workloads and six benchmarks from the PARSEC suite [5] (Table 2(b), (c)) and measure reduction in latency, power savings and IPC improvements over the baseline network configuration.

5. EXPERIMENTAL RESULTS

5.1 Network-only Analysis

Figure 7 (a) shows the load-latency plots of HeteroNoC configurations with the UR traffic pattern. Since the flit sizes and the flits/packet vary across the baseline and HeteroNoC configurations, we measure the latency across packet injection rates rather than flit injection rates. To preserve clarity, we do not show the performance curves for Row2_5+B and Row2_5+BL. However, in Figure 7 (b), we summarize the benefits across all six configurations with respect to the homogeneous design.

As can be seen in Figures 7 (a) and (b), all HeteroNoC configurations outperform the baseline design. The performance improvements with a simple buffer-only redistribution brings an average 9% reduction in latency. Center+B and Diagonal+B configurations, in particular, outperform the Row2_5+B configuration. In Center+B, more buffers are allocated to the central routers and this helps to reduce latency by 10.5% and improve throughput by 11%. The Diagonal+B configuration helps packet flows in the center and corners of the mesh and shows an additional 3% latency reduction and 4% throughput improvement over the Center+B layout. Although Row2_5+B helps reduce latency by reducing the average hop count to a big router, not having big routers at the center of the mesh in this layout, only leads to 4% reduction in latency and 4.5% increase in throughput. This analysis shows that the *placement* of big and small routers is non-trivial and a poor layout choice may lead to network under-performance.

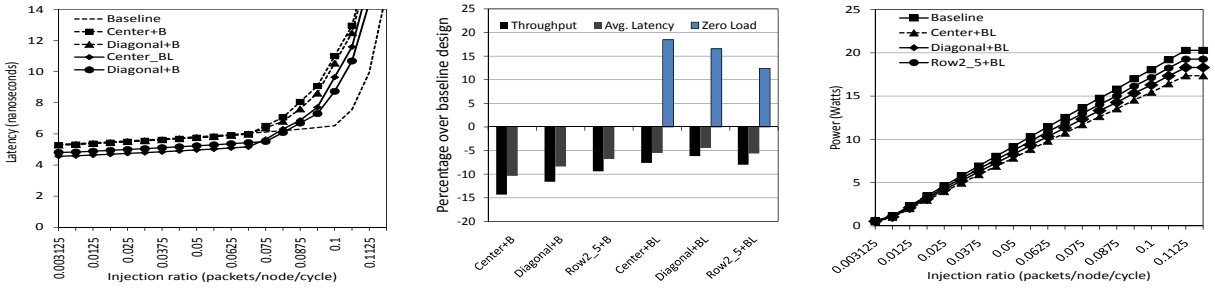
Diagonal+BL is the best HeteroNoC configuration compared to all cases, and reduces latency on an average by 24% and increases throughput by 22% over the baseline design with UR traffic. With combined buffer and link re-distribution in the network, we find, on an average, 12% reduction in zero-load latency with the HeteroNoC configurations. Reducing zero-load latency helps lower end-to-end latency at low loads, and hence, HeteroNoC design helps reduce latency at both low and high injection rates.

The primary reason for the diagonal layouts showing superior performance over their counterparts is that: with diagonal layouts, big routers are distributed over the network and more network flows on an average are able to use these big routers. This shows that, simply allocating more resources to central routers is not optimal, rather, for optimal design, allocation of resources should be such that more flows are able to leverage the performance of big routers (in addition to placing the big routers at locations where utilization is high). Further, allocation of more link resources to big routers complements more buffer allocation in these routers, and hence, all +BL designs outperform their counterpart +B designs.

Figure 7 (c) shows the power curves with the three optimal HeteroNoC designs for UR traffic. HeteroNoC reduces power by balancing the number of big and small routers in the network. Buffer only redistribution does not reduce the overall power in the network significantly compared to baseline design (hence omitted in all the power plots), since, with this kind of redistribution, neither crossbar width nor buffer resources are reduced in the network. However, with the combined buffer and link re-distribution, the total buffer resources are reduced by 33% (shown in Table 1). Additionally, in HeteroNoC design, there are 48 small routers and 16 big routers compared to all 64 medium sized routers in the baseline design. Many small routers help cut down the power consumption further. On an average, we find 21.5% (28% with Diagonal+BL) power reduction for UR traffic across all HeteroNoC designs with the combined buffer and link re-distribution.

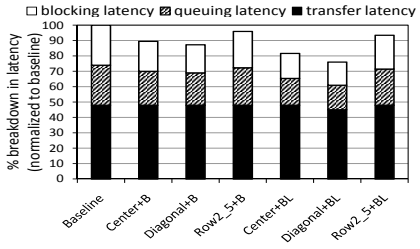
The detailed latency and power breakdowns across all HeteroNoC configurations with UR traffic are depicted in Figure 8. The network latency is divided into transfer latency, queuing delay at the source node, and blocking delay at intermediate hops. The overall power consumption in the routers is composed of the power consumptions in the router buffers (to read/write), arbiters and logic, crossbar transfer and link transmission. With the HeteroNoC design, latency reduction results primarily by reducing the queuing and blocking latency (Figure 8 (a)). Wider links in big routers and combining two flits for simultaneous transmission help reduce these latency components. Power reduction comes primarily from reduction in buffers (33%) and crossbar power (Figure 8 (b)).

Apart from UR traffic pattern, we also analyzed HeteroNoC configurations with transpose, bit-complement and self-similar traffic patterns (not shown here due to space limitations) and observed that the load-latency and power consumption curves are very similar in trend to those obtained with UR traffic. This re-enforces the fact that our HeteroNoC design is not limited to a particular traffic

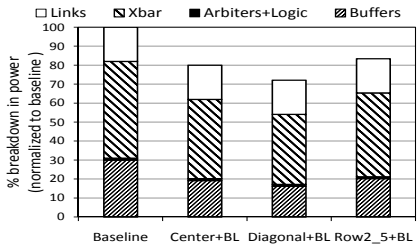


(a) Latency (b) Throughput improvement and latency reduction (c) Power

Figure 9: Performance and network power behavior with NN traffic.



(a) Latency breakdown



(b) Power breakdown

Figure 8: Latency and power breakdown with UR traffic.

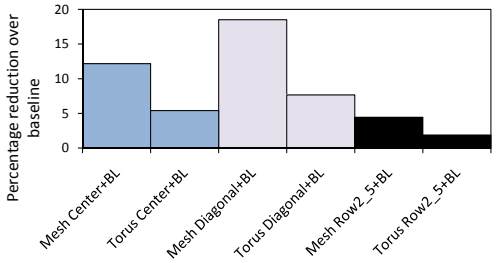


Figure 10: Latency reduction in an 8x8 mesh vs. torus.

pattern or layout. Rather, by redistributing resources intelligently, HeteroNoC is able to provide power-performance benefits across various traffic patterns. The only anomaly we found was with the NN traffic pattern (Figure 9 shows the performance-power curves with NN traffic). With NN pattern, communication occurs between neighboring nodes. Since in HeteroNoC designs, we reduce the buffer and link resources in most of the network routers, this affects performance for neighboring router communications and the network saturates earlier compared to the baseline design. Although, the zero-load latency reduces by 16% on an average in the combined buffer and link redistribution designs, the average network latency increases by 7% and throughput reduces by 9.5%. Power benefits are also minimal (7%) with NN traffic. With NN traffic, Center+BL performs better than Diagonal+BL, since having *all* big routers in the center aids nearest neighbor communication between the central routers.

5.1.1 Comparison with an Edge-Symmetric Network: Torus

To examine how our proposal performs with an edge-symmetric network, we evaluated our design with an 8x8 torus network. Figure 10 shows the results of this analysis, where the bars show the average latency reduction across all the application benchmarks summarized in Table 2. For comparison, we also show the results for a mesh network. While using heterogeneous routers in a torus topology provides marginal performance gain, the benefits are on an average 44% less compared to employing heterogeneity in a mesh topology. Intuitively, this makes sense since on an average, roughly 50% traffic use the loop-around links in the torus network and these traffic do not benefit from the additional resources solely at the central routers in the torus. Also, unlike the mesh topology, the heterogeneity in resource consumption is not primarily restricted to the center of the network, and changes dynamically with applications. Overall, static allocation of more resources to a few routers in a torus does not provide significant benefits.

5.2 System-level Analysis

For system-level analysis, we evaluate our proposed CMP network configurations using a trace-driven, cycle-accurate x86 CMP simulator. The dynamic memory traces are collected using Simics. Our trace format consists of load/stores and the number of non-memory instructions between them. Our CPU model consists of a fetch/issue/commit out-of-order pipeline with a reorder buffer (Table 2). We use a 64-tile CMP laid out on an 8x8 mesh platform with each tile consisting of a core with a private write-back L1 cache and an L2 cache bank. The memory instructions are modeled through the detailed memory hierarchy and network model. The memory hierarchy uses a two-level directory-based MESI cache coherence protocol and the network connects the cores, L2 cache banks, and memory controllers. All requests and responses are faithfully modeled by the network. The CPU, network configurations and workload details are given in Table 2, and the VC configurations for the two networks (homogeneous and HeteroNoC) are given in Table 1.

Figure 11 (a) shows the impact of using various HeteroNoC layouts on network latency. Re-distributing resources appropriately helps reduce contention in the network and reduces end-to-end delay. Simultaneous transmission of flits reduces serialization latency and thus, transfer latency (shown in Figure 11 (b)). Overall, we find 18.5% average latency reduction with the Diagonal+BL layout

Table 2: CPU, Cache, Network and Workloads Configuration

CPU and network configuration
Processor Pipeline: 64 x86 based 2.2 GHz (nominal) processors, two-way out of order, 64-entry instruction window
Fetch/Exec/Commit width: 3 instructions per cycle in each core; only 1 can be a memory operation
L1 Caches: 32 KB per-core(private), 4-way set associative, 128B block size, 2-cycle latency, write-back, split I/D caches
L2 Caches: 1MB bank(per-core), shared, 16-way set associative, 128B block size, 6-cycles bank latency, 32 MSHRs
Main Memory: 4GB DRAM, up to 16 outstanding requests for each processor, 400 cycle access, 4 memory controllers placed at corners of the mesh network (baseline)
Network and Router: 8x8 mesh network(each tile consists of 1 CPU, private L1 cache, 1 shared L2 bank and 1 router), 2-stage wormhole switched router, X-Y routing.

(a)

Commercial workloads
App. Benchmark(SAP): SAP stands for Standard Application Benchmark and represents a sales and distribution workload. **Server Workload(SPECjbb)**: SPECjbb2000 is a Java based benchmark that models a 3-tier system. We simulated 64 warehouses on 64 processors and start measurements 30 seconds after ramp-up time. **Transaction Processing(TPC-C)**: TPC-C is an OLTP benchmark.TPC-C simulates a complete computing environment where a population of users executes transactions against a database. **SPEC Java App. Server(SJAS)**: SJAS is a multi-tier benchmark for measuring the performance of J2EE technology-based application servers. The traces for TPC-C, SAP and SJAS benchmark were collected from a CMP server configurations at Intel Corporation and we use 64 threads from each benchmark.
SPEC2K6 libquantum benchmark: libquantum simulates a quantum computer, running Shor's polynomial time factorization algorithm. We use liquantum when co-evaluating HeteroNoC with asymmetric CMPs.

(b)

PARSEC
PARSEC suite includes emerging RMS applications as well as large scale multi-threaded programs for CMPs. From this suite we choose three application benchmarks(**ferret (frrt), facesim (fsim) and vips**) and three kernel benchmarks(**canneal (canl), dedup (ddup) and streamcluster (sclust)**) and ran them with simlarge input sets. Traces were collected on a CMP platform(see config. in (a)) using Simics for 64 threads of each benchmark for 20 million L2 references and then simulated in our cycle-accurate processor-cache-network simulator.

(c)

across all of our application suites. Figures 11 (c) and (d) show the reduction in power consumption and power breakdown, respectively. Reduction in crossbar power (using 128b wires vs 192b wires in baseline) and buffer power (since HeteroNoC uses 33% fewer buffers) are the primary reasons for reduction in overall network power. On an average, we find 18% (22% with Diagonal+BL) network power reduction with HeteroNoC designs across all application suites we evaluated.

Figures 12 (a) and (b) depict the percentage improvement in IPC over the baseline case, and again Diagonal+BL has the best results with 12% and 10% average improvements in IPC for commercial applications and PARSEC benchmarks, respectively. Thus, with the HeteroNoC designs, we not only use 33% fewer buffer and save in network power consumption, but also see IPC improvement.

6. CASE STUDY I: EVALUATION WITH MEMORY CONTROLLERS

A recent work by Abts et al. in [2] showed the efficacy of an optimal memory controller arrangement in a mesh based CMP platform with X-Y routing. They considered several memory controller placements and showed that when multiple memory controllers are placed either along the network diagonals or in a diamond shape in the network, the average packet latency and request-response variance is minimized. Since, this work considers the *placements* of big and small routers in a mesh network, we co-examine Abts et al.'s memory controller layouts with our HeteroNoC designs.

For this co-evaluation, we considered both diagonal and diamond layouts of memory controllers in the mesh network (the best layouts proposed in [2]) with the big routers placed along the network diagonals (the best HeteroNoC layout). Diamond placement of memory controllers complements the diagonal placement of big routers (Diagonal+BL design) with the memory controllers distributed uniformly and symmetrically in the network. The number of memory controllers used in this analysis is 16 (two controllers per row/column of the mesh). With diagonal placement of memory controllers, the big routers have additional local port VC buffers that relay traffic destined to memory controllers. On an L2 miss, we use the low order address bits above the cache line address to choose the destination memory controller.

We measured the round trip request-response delay from when a memory request is generated by a core to when the request arrives back at the core after being serviced by the memory controller and DRAM. To model the behavior of Miss Status Handling Registers (MSHRs) when using UR traffic, we perform a closed-loop evaluation, where up to 16 outstanding requests are allowed from a node before halting packet injection in that node until the response packets arrive from the requested memory controller. With benchmarks, we use the core and network configurations mentioned in Table 2 (a).

Figure 13 (a) shows the reduction in request-response latency for the three scenarios evaluated - (a) *diamond* memory controller placement in a *homogeneous* network where all routers have 3VCs and 192b links; annotated as *Diamond_homoNoC* in the figure, (b) *diamond* memory controller placement in a *heterogeneous* network (Diagonal+BL design); annotated as *Diamond_heteroNoC*, and (c) *diagonal* memory controller placement in the *heterogeneous* network (Diagonal+BL design); annotated as *Diagonal_heteroNoC*.

Diamond_homoNoC is the design which Abts et al. evaluated in their work. Compared to this design that reduces the average round trip latency by 8%, Diamond_heteroNoC helps lower average request-response latency across all workloads by 22%. This is solely due to the HeteroNoC design that provisions more resources to the routers that service more traffic. Diagonal_heteroNoC performs the best and reduces average round trip delay by 28%. With Diagonal_heteroNoC design, the memory controllers are attached to big routers, which have more resources to effectively handle increased traffic without getting congested.

In addition, a diagonal placement of memory controllers helps reduce the variance (or jitter) in the request latency to memory controllers. Figure 13 (b) shows the request latency vs. standard deviation in request latency for the three configurations. The Diagonal_heteroNoC design reduces the variance (0.46) when compared to the Diamond_homoNoC design (0.66). A lower standard deviation together with reduced latency between cores and memory controllers indicates that a combination of HeteroNoC and the diagonal configuration of memory controllers, provides predictable delays to memory controllers regardless of which core a thread is scheduled on.

7. CASE STUDY II: EVALUATION WITH ASYMMETRIC CORES

Recent research has shown the potential of heterogeneous CMPs in providing high single-threaded performance when thread parallelism is low, and high throughput when thread parallelism is

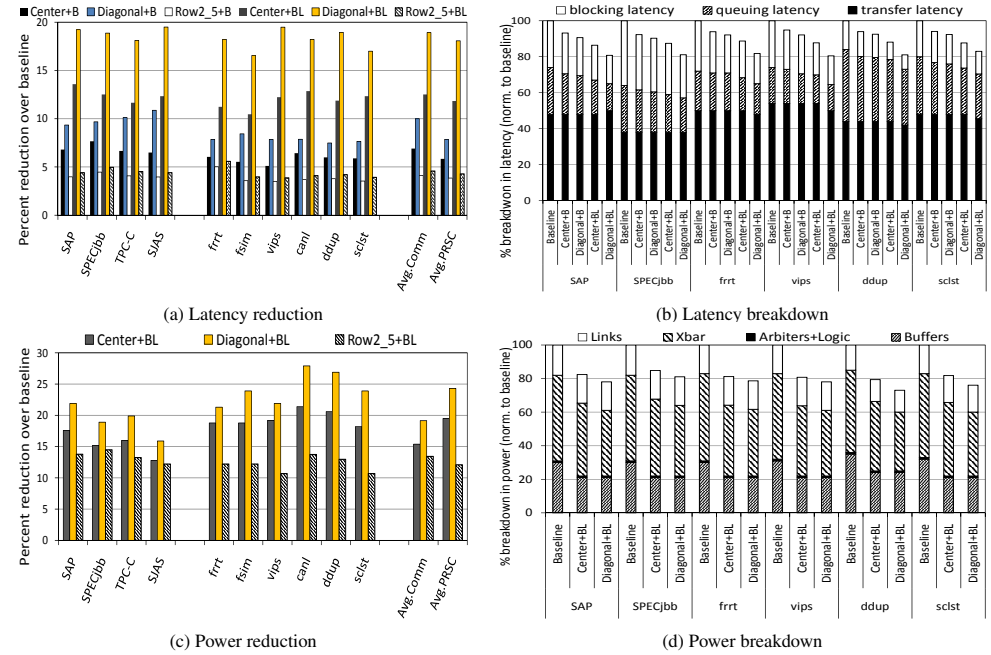


Figure 11: Latency and power reduction with applications.

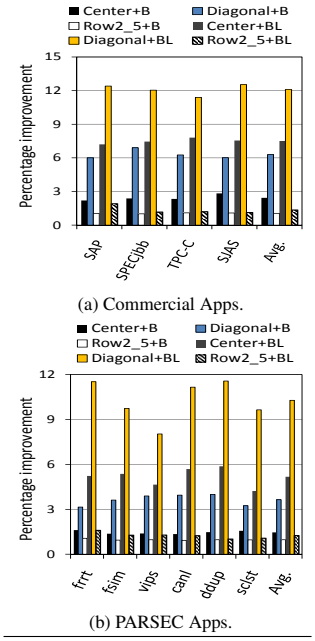


Figure 12: IPC improvement.

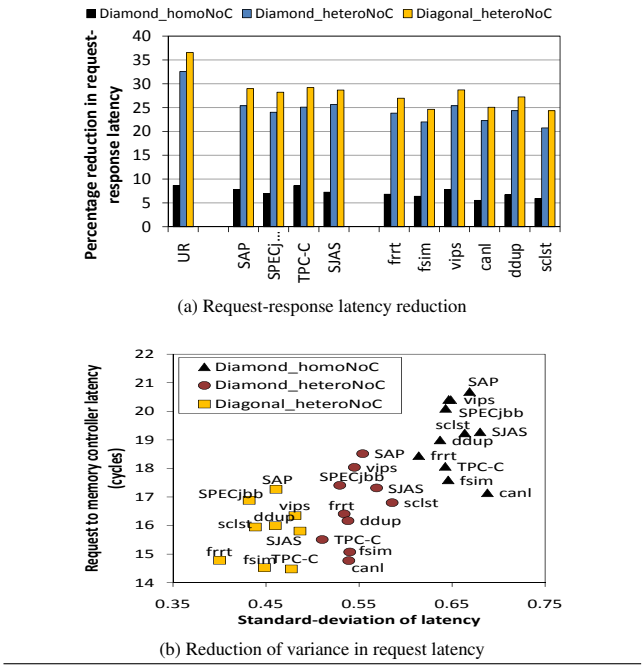


Figure 13: Results with memory-controller placements.

high [11,18,19,23]. They do so by hosting the low thread-level parallel (TLP) application on a powerful core and the high TLP application on many simple cores. Most prior works on heterogeneous CMPs have focused on the mapping/scheduling of applications to cores [18], using a single-ISA heterogeneous platform [19] and analyzing energy trade-offs of asymmetric platforms [23]. However, no prior work has studied the impact of the on-chip interconnect in a heterogeneous CMP platform. Clearly, with asymmetric cores sharing the same CMP substrate, the interconnect will cater to different demands - the high TLP applications may have significant on-chip shared traffic and can *tolerate* network delays, while the latency sensitive applications require a *low latency* interconnect.

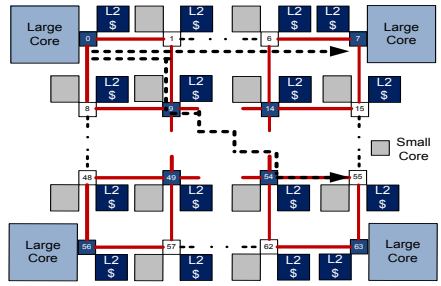
We envision an asymmetric CMP platform to have a heterogeneous interconnect. Figure 14 (a) shows the asymmetric CMP and the HeteroNoC platform used in our evaluation. The large cores are multiple-issue out-of-order powerful cores with configuration shown in Table 2 (a) and the small cores are single-issue in-order cores with similar private cache configuration as the large cores. We assume a large core to occupy 4x the area of a small core and consider an asymmetric CMP platform consisting of 4 large cores placed at the corners of the heterogeneous mesh network (Diagonal+BL) and 60 small cores placed at the remaining nodes of the mesh network. The reason for placing the large cores far apart is that these cores will consume the most power and host single threaded applications that require minimal communication with other large cores.

To show the impact of a heterogeneous network in an asymmetric platform, we evaluate three scenarios - (a) *HomoNoC-XY*, where all routers in the network have 3 VCs/PC and 192b link, (b) *HeteroNoC-XY*, where we use a Diagonal+BL HeteroNoC design and use X-Y routing, and (c) *HeteroNoC-Table+XY*, where we use two routing protocols - table based routing for traffic originating from and destined to large cores and X-Y routing for the remaining small cores. The table based routing leverages the performance of big routers in the heterogeneous network to aid the latency critical nature of packets originating from (and destined to) large cores.

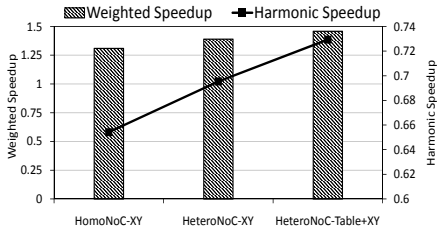
With table-based routing, packets to/from large cores are routed such that they leverage the big routers maximally. Figure 14 (a) shows two paths that packets originating from core 0 use to reach cache banks connected to routers 7 and 55. When packets originating from core 0 (and hence router 0) want to reach a cache bank connected to router 7, they are routed similar to X-Y routing. However, when these packets want to access a cache bank connected to router 55, they are routed in a zig-zag X-Y-X-Y fashion so as to maximally use the big routers laid out along the diagonals. Table-based routing mandates maintaining source/destination pair information in all routers in the network. However, we use table-based routing only for packets to/from the four large cores, and hence, the table sizes are minimal when compared to maintaining table information for all 64 cores in the network. Since the table-based routing approach can lead to deadlock scenarios, reserved escape VCs are used in the big routers to resolve deadlocks. We schedule 60 threads of SPECjbb on the small cores and one instance of libquantum benchmark on each large core. libquantum is a latency sensitive benchmark, whereas the multi-threaded SPECjbb, with each thread simulating one instance of a warehouse in a 3-tier JAVA server, represents a throughput-oriented application having high TLP.

Figure 14 (b) shows the weighted speedup and harmonic speedup of the system with the three network layouts. These speedup metrics are two commonly used multi-program performance metrics [8] based on comparing the IPC of an application when it runs alone versus when it runs together with others. Harmonic speedup is a measure of both performance and fairness. When computing the harmonic speedup, we take into account the IPC of the slowest thread of SPECjbb for computing the metric.

Table-based routing expedites libquantum packets by routing them through big routers. It also helps to accelerate SPECjbb packets by reducing contention in small routers, since small routers are now less frequently used by libquantum packets. Table-based routing along with X-Y routing in a heterogeneous network provides the maximum weighted speedup demonstrating the advantages of selectively expediting libquantum packets. HeteroNoC+XY and HeteroNoC-Table+XY show 6% and 11% improvement, respectively, in weighted speedup over the HomoNoC-XY design. The harmonic speedup also increases with HeteroNoC and table based-routing showing that selective expedition of libquantum packets does not lead to system unfairness. With HeteroNoC-Table+XY, we find 11.5% improvement in harmonic speedup over the HomoNoC-XY design.



(a) Layout of an asymmetric CMP



(b) Performance improvement of SPECjbb and libquantum

Figure 14: Evaluation with an asymmetric CMP.

8. RELATED WORK

Interest in on-chip networks has rapidly garnered momentum over the last few years. Towards this end, most research has focused on two major themes - improving the performance and reducing power consumption. For improving performance, researchers have proposed the use of VCs and path speculation [27], smart pipelines [25], dynamic traffic re-distribution [16], and novel network topologies [14, 15]. To reduce power and thermal profiles in on-chip networks, several prior works have proposed communication aware topologies [7, 9], power and thermal management in routers [12, 22, 28], and bufferless routing [24]. Our approach is different from all these proposals since, we leverage the fact that networks have non-uniform resource usage, and hence, allocate resources accordingly without changing the routing or the traffic flows.

Prior work in the SoC domain have proposed customization of on-chip resources like buffers [13], links [10, 20, 26] and topology [20, 21]. However, all these works adopt a static approach, where optimal buffer sizes and link widths are pre-determined at design time based on a detailed analysis of application-specific traffic patterns. The sizing is optimal for only one particular application or similar application suites and one hardware mapping. Thus, application characteristic needs to be known a-priori and the network is customized for each target application. In contrast, we propose a truly heterogeneous interconnect that is independent of application type and is targeted for general purpose CMPs, which host a variety of applications, and, where customizing the NoC for one application is prohibitive.

A recent work by Bakhoda et al. [3] showed the effectiveness of using two kinds of routers – limited and full connectivity routers, in a mesh-based NoC for GPGPU applications. This design is based on the observation of many-to-few-to-many traffic pattern in many-core accelerators. The limited connectivity routers allow traffic to be routed only in certain directions and hence is inefficient (as identified by the authors) for handling large number of cache to cache transfers (e.g. coherence messages). Thus, such limited connectivity and restricted routing works well in the context of accelerators, and it is not effective for general purpose CMPs, where traffic patterns are not deterministic and there are large number of inter-cache messages. On the other hand, HeteroNoC is a generic concept that can be exploited for improving performance and power savings in any non-edge symmetric NoC.

Overall, we believe that no prior work has made a case for using heterogeneous network for general purpose CMPs with careful consideration to the *types*, *number* and *placement* of heterogeneous routers.

9. CONCLUSIONS

In this paper, we propose to leverage the non-uniform resource usage in a homogeneous mesh interconnect for designing a heterogeneous network, composed of big and small routers, by redistributing the buffer and link bandwidth. We explore the design space analysis in choosing the *size, number* and *placement* of these big and small routers and show that our HeteroNoC design with big routers placed along the network diagonals performs significantly better than the traditional homogeneous network under a variety of

traffic patterns. Detailed experimental analysis shows that for synthetic traffic patterns, our HeteroNoC design with the big routers along the diagonals provides maximum benefits (23% and 26% reduction in latency and power consumption, respectively) compared to an equivalent homogeneous network. Evaluation with 10 diverse application benchmarks shows that HeteroNoC can provide IPC improvements up to 12% and power reduction up to 22%. Additionally, our analysis shows that the non-uniformity in resource utilization is an artifact of any non-edge symmetric network design and deterministic X-Y routing, and this inherent artifact can be leveraged to better customize the network.

In addition, we also demonstrated how the HeteroNoC can be useful for enhancing the performance of heterogeneous CMPs and complement the placement of memory controllers in an NoC. Based on our evaluations, we argue in favor of using heterogeneous NoCs for enhancing the performance and reducing power consumption of future general purpose multicore architectures, specifically designed with non-edge symmetric networks. The design is simple, should be applicable to a wide class of network configurations and does not incur significant overheads for practical implementation.

10. ACKNOWLEDGEMENTS

We thank Ravi Iyer (Intel Labs/IPR) for his valuable suggestions in improving our work. We also thank the anonymous reviewers for their reviews and comments towards improving this paper. This work is supported in part by National Science Foundation (NSF) grants CCF-0702617, CNS-0916887 and CCF-0903432.

11. REFERENCES

[1] International Technology Roadmap for Semiconductors, 2010 Edition. http://www.itrs.net/Links/2010ITRS/Home2010.htm.

[2] D. Abts, N. D. Enright Jerger, J. Kim, D. Gibson, and M. H. Lipasti. Achieving Predictable Performance Through Better Memory Controller Placement in Many-Core CMPs. In *36th ISCA, 2009.*

[3] A. Bakhoda, J. Kim, and T. M. Aamodt. Throughput-Effective On-Chip Networks for Manycore Accelerators. In *MICRO-43, 2010.*

[4] J. Balfour and W. J. Dally. Design Tradeoffs for Tiled CMP On-Chip Networks. In *ICS-20*, 2006.

[5] C. Bienia, S. Kumar, J. P. Singh, and K. Li. The PARSEC Benchmark Suite: Characterization and Architectural Implications. In *17th PACT*, 2008.

[6] W. J. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2003.

[7] R. Das, S. Eachempati, A. Mishra, V. Narayanan, and C. Das. Design and Evaluation of a Hierarchical On-Chip Interconnect for Next-Generation CMPs. In *15th HPCA, 2009.*

[8] S. Eyerman and L. Eeckhout. System-Level Performance Metrics for Multiprogram Workloads. *Micro, IEEE*, 2008.

[9] B. Grot, J. Hestness, S. Keckler, and O. Mutlu. Express Cube Topologies for on-Chip Interconnects. In *15th HPCA 2009*.

[10] Z. Guz, I. Walter, E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny. Network Delays and Link Capacities in Application-Specific Wormhole NoCs. *VLSI '07: Journal of VLSI Design, vol. 2007*.

[11] M. D. Hill and M. R. Marty. Amdahl's Law in the Multicore Era. *IEEE COMPUTER*, 2008.

[12] Y. Hoskote, S. Vangal, A. Singh, N. Borkar, and S. Borkar. A 5-GHz Mesh Interconnect for a Teraflops Processor. In *Micro, IEEE*, volume 27, pages 51–61, Sept.-Oct. 2007.

[13] J. Hu and R. Marculescu. Application-specific buffer space allocation for networks-on-chip router design. In *ICCAD*, 2004.

[14] J. Kim, W. Dally, S. Scott, and D. Abts. Technology-Driven, Highly-Scalable Dragonfly Topology. In *35th ISCA 2008*.

[15] J. Kim, W. J. Dally, and D. Abts. Flattened Butterfly: A Cost-Efficient Topology for High-Radix Networks. In *34th ISCA*, 2007.

[16] J. Kim, D. Park, T. Theocharides, N. Vijaykrishnan, and C. R. Das. A Low Latency Router Supporting Adaptivity for On-Chip Router. In *42nd DAC*, 2005.

[17] A. Kumar, L.-S. Peh, P. Kundu, and N. K. Jha. Express Virtual Channels: Towards the Ideal Interconnection Fabric. In *34th ISCA*, 2007.

[18] R. Kumar, D. M. Tullsen, N. P. Jouppi, and P. Ranganathan. Heterogeneous Chip Multiprocessors. *Computer*, 38(11), 2005.

[19] R. Kumar, D. M. Tullsen, P. Ranganathan, N. P. Jouppi, and K. I. Farkas. Single-ISA Heterogeneous Multi-Core Architectures for Multithreaded Workload Performance. *SIGARCH Computer Architecture News*, 32(2), 2004.

[20] R. Marculescu, U. Y. Ogras, and N. H. Zamora. Computation and Communication Refinement for Multiprocessor SoC Design: A System-Level Perspective. *ACM TODAES*, 11(3), 2006.

[21] A. Mejia, M. Palesi, J. Flich, S. Kumar, P. López, R. Hoismark, and J. Duato. Region-Based Routing: A Mechanism to Support Efficient Routing Algorithms in NoCs. *IEEE Transactions On VLSI Systems*, 17(3), 2009.

[22] A. Mishra, R. Das, S. Eachempati, R. Iyer, V. Narayanan, and C. Das. A Case for Dynamic Frequency Tuning in On-Chip Networks. In *MICRO-42*, 2010.

[23] T. Y. Morad, U. C. Weiser, A. Kolodny, M. Valero, and E. Ayguade. Performance, Power Efficiency and Scalability of Asymmetric Cluster Chip Multiprocessors. *IEEE Computer Architecture Letters*, 2006.

[24] T. Moscibroda and O. Mutlu. A Case for Bufferless Routing in On-Chip Networks. In *36th ISCA*, 2009.

[25] R. Mullins, A. West, and S. Moore. Low-Latency Virtual-Channel Routers for On-Chip Networks. In *31st ISCA*, 2004.

[26] S. Murali, M. Coenen, A. Radulescu, K. Goossens, and G. De Micheli. Mapping and Configuration Methods for Multi-Use-Case Networks on Chips. In *ASPDAC*, 2006.

[27] L.-S. Peh and W. J. Dally. A Delay Model and Speculative Architecture for Pipelined Routers. In *7th HPCA*, 2001.

[28] L. Shang, L.-S. Peh, A. Kumar, and N. K. Jha. Thermal Modeling, Characterization and Management of On-Chip Networks. In *MICRO-37*, 2004.

[29] H. Wang, L.-S. Peh, and S. Malik. A Technology-Aware and Energy-Oriented Topology Exploration for On-Chip Networks. In *DATE*, 2005.

[30] H. Wang, X. Zhu, L.-S. Peh, and S. Malik. Orion: A Power-Performance Simulator for Interconnection Networks. In *MICRO-35*, 2002.